# Descripción del problema

Se detectó que la entrada del usuario se utiliza manualmente para construir una cadena SQL, lo cual expone el sistema a vulnerabilidades de inyección SQL.

# Impacto:

Este problema es crítico ya que permite a los atacantes ejecutar comandos maliciosos directamente en la base de datos, lo que puede resultar en pérdida o corrupción de datos, filtración de información confidencial, e incluso comprometer todo el sistema.

# Ubicación del Código:

Archivo: Desarrollo/MentalSync/Fuentes/server/controllers/paciente.controller.js Línea: 117

# Solución Sugerida:

Para solucionar este problema, se debe evitar la concatenación directa de entradas de usuario en las consultas SQL. Utiliza un ORM (como Sequelize) o sentencias preparadas con parámetros para manejar las entradas de usuario de forma segura. Ejemplo de solución utilizando sentencias preparadas:

const query = 'UPDATE paciente SET column\_name = ? WHERE condition = ?'; const values = [value1, value2];

db.execute(query, values);

# Referencias:

* 1. OWASP: SQL Injection Prevention Cheat Sheet
  2. Documentación de Sequelize para consultas parametrizadas

# Descripción del problema

Se encontró una configuración que deshabilita la verificación de certificados TLS mediante la variable 'NODE\_TLS\_REJECT\_UNAUTHORIZED' configurada como '0'. Esto permite la conexión a servidores no seguros.

# Impacto:

Este problema es de severidad media, ya que expone el sistema a posibles ataques Man- In-The-Middle (MITM), donde los atacantes pueden interceptar y modificar la comunicación entre el cliente y el servidor. También debilita la confianza en la seguridad de las conexiones HTTPS.

# Ubicación del Código:

Archivo: Desarrollo/MentalSync/Fuentes/server/src/db.js Línea: 25

# Solución Sugerida:

Evita deshabilitar la verificación TLS a menos que sea estrictamente necesario y solo en entornos de desarrollo. Si se requiere en producción, considera usar certificados válidos y confiables.

# Referencias:

* 1. Node.js Documentation: Using TLS in Node.js
  2. OWASP: Transport Layer Protection Cheat Sheet